*장현주\_2023011923*

|  |
| --- |
| **2024 ALTIS SW 개인 활동 보고서** |

|  |  |
| --- | --- |
| **활동 개요** | MPU6050 3D 시각화 구현 |
| **일시** | 2024년 5월 12일 00시 |
| **작성자** | 김민주, 장현주 |
| **활동 내용** | *※ 어떤 활동을 진행하였는지 자세히 쓰기*  (opengl과 processing을 사용하여 보았습니다.)  - processing 파일을 다운받고, ‘toxi’라이브러리를 다운받아 processing의 라이브러리 파일안에 복사하여 넣습니다.  - 코드는 아두이노 파일 안에 있는 예제에서 몇 가지 수정하여(밑에 적힌 코드로) 저장한 뒤에 processing을 실행할 때 아두이노 파일이 아닌 processing파일을 실행한 뒤에 실행하면 실행 화면이 작동된다.  *※ 코드가 있다면 어떤 코드인지 + 코드에 주석 달기*  // I2C device class (I2Cdev) demonstration Processing sketch for MPU6050 DMP output  // 6/20/2012 by Jeff Rowberg <jeff@rowberg.net>  // Updates should (hopefully) always be available at https://github.com/jrowberg/i2cdevlib  //  // Changelog:  // 2012-06-20 - initial release  /\* ============================================  I2Cdev device library code is placed under the MIT license  Copyright (c) 2012 Jeff Rowberg  Permission is hereby granted, free of charge, to any person obtaining a copy  of this software and associated documentation files (the "Software"), to deal  in the Software without restriction, including without limitation the rights  to use, copy, modify, merge, publish, distribute, sublicense, and/or sell  copies of the Software, and to permit persons to whom the Software is  furnished to do so, subject to the following conditions:  The above copyright notice and this permission notice shall be included in  all copies or substantial portions of the Software.  THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR  IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,  FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE  AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER  LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,  OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN  THE SOFTWARE.  ===============================================  \*/  import processing.serial.\*;  import processing.opengl.\*;  import toxi.geom.\*;  import toxi.processing.\*;  // NOTE: requires ToxicLibs to be installed in order to run properly.  // 1. Download from https://github.com/postspectacular/toxiclibs/releases  // 2. Extract into [userdir]/Processing/libraries  // (location may be different on Mac/Linux)  // 3. Run and bask in awesomeness  ToxiclibsSupport gfx;  Serial port; // The serial port  char[] teapotPacket = new char[14]; // InvenSense Teapot packet  int serialCount = 0; // current packet byte position  int synced = 0;  int interval = 0;  float[] q = new float[4];  Quaternion quat = new Quaternion(1, 0, 0, 0);  float[] gravity = new float[3];  float[] euler = new float[3];  float[] ypr = new float[3];  void setup() {  // 300px square viewport using OpenGL rendering  size(300, 300, OPENGL);  gfx = new ToxiclibsSupport(this);  // setup lights and antialiasing  lights();  smooth();    // display serial port list for debugging/clarity  println(Serial.list());  // get the first available port (use EITHER this OR the specific port code below)  String portName = Serial.list()[0];    // get a specific serial port (use EITHER this OR the first-available code above)  //String portName = "COM4";    // open the serial port  port = new Serial(this, portName, 115200);    // send single character to trigger DMP init/start  // (expected by MPU6050\_DMP6 example Arduino sketch)  port.write('r');  }  void draw() {  if (millis() - interval > 1000) {  // resend single character to trigger DMP init/start  // in case the MPU is halted/reset while applet is running  port.write('r');  interval = millis();  }    // black background  background(0);    // translate everything to the middle of the viewport  pushMatrix();  translate(width / 2, height / 2);  // 3-step rotation from yaw/pitch/roll angles (gimbal lock!)  // ...and other weirdness I haven't figured out yet  //rotateY(-ypr[0]);  //rotateZ(-ypr[1]);  //rotateX(-ypr[2]);  // toxiclibs direct angle/axis rotation from quaternion (NO gimbal lock!)  // (axis order [1, 3, 2] and inversion [-1, +1, +1] is a consequence of  // different coordinate system orientation assumptions between Processing  // and InvenSense DMP)  float[] axis = quat.toAxisAngle();  rotate(axis[0], -axis[1], axis[3], axis[2]);  // draw main body in red  fill(255, 0, 0, 200);  box(10, 10, 200);    // draw front-facing tip in blue  fill(0, 0, 255, 200);  pushMatrix();  translate(0, 0, -120);  rotateX(PI/2);  drawCylinder(0, 20, 20, 8);  popMatrix();    // draw wings and tail fin in green  fill(0, 255, 0, 200);  beginShape(TRIANGLES);  vertex(-100, 2, 30); vertex(0, 2, -80); vertex(100, 2, 30); // wing top layer  vertex(-100, -2, 30); vertex(0, -2, -80); vertex(100, -2, 30); // wing bottom layer  vertex(-2, 0, 98); vertex(-2, -30, 98); vertex(-2, 0, 70); // tail left layer  vertex( 2, 0, 98); vertex( 2, -30, 98); vertex( 2, 0, 70); // tail right layer  endShape();  beginShape(QUADS);  vertex(-100, 2, 30); vertex(-100, -2, 30); vertex( 0, -2, -80); vertex( 0, 2, -80);  vertex( 100, 2, 30); vertex( 100, -2, 30); vertex( 0, -2, -80); vertex( 0, 2, -80);  vertex(-100, 2, 30); vertex(-100, -2, 30); vertex(100, -2, 30); vertex(100, 2, 30);  vertex(-2, 0, 98); vertex(2, 0, 98); vertex(2, -30, 98); vertex(-2, -30, 98);  vertex(-2, 0, 98); vertex(2, 0, 98); vertex(2, 0, 70); vertex(-2, 0, 70);  vertex(-2, -30, 98); vertex(2, -30, 98); vertex(2, 0, 70); vertex(-2, 0, 70);  endShape();    popMatrix();  }  void serialEvent(Serial port) {  interval = millis();  while (port.available() > 0) {  int ch = port.read();  if (synced == 0 && ch != '$') return; // initial synchronization - also used to resync/realign if needed  synced = 1;  print ((char)ch);  if ((serialCount == 1 && ch != 2)  || (serialCount == 12 && ch != '\r')  || (serialCount == 13 && ch != '\n')) {  serialCount = 0;  synced = 0;  return;  }  if (serialCount > 0 || ch == '$') {  teapotPacket[serialCount++] = (char)ch;  if (serialCount == 14) {  serialCount = 0; // restart packet byte position    // get quaternion from data packet  q[0] = ((teapotPacket[2] << 8) | teapotPacket[3]) / 16384.0f;  q[1] = ((teapotPacket[4] << 8) | teapotPacket[5]) / 16384.0f;  q[2] = ((teapotPacket[6] << 8) | teapotPacket[7]) / 16384.0f;  q[3] = ((teapotPacket[8] << 8) | teapotPacket[9]) / 16384.0f;  for (int i = 0; i < 4; i++) if (q[i] >= 2) q[i] = -4 + q[i];    // set our toxilibs quaternion to new data  quat.set(q[0], q[1], q[2], q[3]);  /\*  // below calculations unnecessary for orientation only using toxilibs    // calculate gravity vector  gravity[0] = 2 \* (q[1]\*q[3] - q[0]\*q[2]);  gravity[1] = 2 \* (q[0]\*q[1] + q[2]\*q[3]);  gravity[2] = q[0]\*q[0] - q[1]\*q[1] - q[2]\*q[2] + q[3]\*q[3];    // calculate Euler angles  euler[0] = atan2(2\*q[1]\*q[2] - 2\*q[0]\*q[3], 2\*q[0]\*q[0] + 2\*q[1]\*q[1] - 1);  euler[1] = -asin(2\*q[1]\*q[3] + 2\*q[0]\*q[2]);  euler[2] = atan2(2\*q[2]\*q[3] - 2\*q[0]\*q[1], 2\*q[0]\*q[0] + 2\*q[3]\*q[3] - 1);    // calculate yaw/pitch/roll angles  ypr[0] = atan2(2\*q[1]\*q[2] - 2\*q[0]\*q[3], 2\*q[0]\*q[0] + 2\*q[1]\*q[1] - 1);  ypr[1] = atan(gravity[0] / sqrt(gravity[1]\*gravity[1] + gravity[2]\*gravity[2]));  ypr[2] = atan(gravity[1] / sqrt(gravity[0]\*gravity[0] + gravity[2]\*gravity[2]));    // output various components for debugging  //println("q:\t" + round(q[0]\*100.0f)/100.0f + "\t" + round(q[1]\*100.0f)/100.0f + "\t" + round(q[2]\*100.0f)/100.0f + "\t" + round(q[3]\*100.0f)/100.0f);  //println("euler:\t" + euler[0]\*180.0f/PI + "\t" + euler[1]\*180.0f/PI + "\t" + euler[2]\*180.0f/PI);  //println("ypr:\t" + ypr[0]\*180.0f/PI + "\t" + ypr[1]\*180.0f/PI + "\t" + ypr[2]\*180.0f/PI);  \*/  }  }  }  }  void drawCylinder(float topRadius, float bottomRadius, float tall, int sides) {  float angle = 0;  float angleIncrement = TWO\_PI / sides;  beginShape(QUAD\_STRIP);  for (int i = 0; i < sides + 1; ++i) {  vertex(topRadius\*cos(angle), 0, topRadius\*sin(angle));  vertex(bottomRadius\*cos(angle), tall, bottomRadius\*sin(angle));  angle += angleIncrement;  }  endShape();    // If it is not a cone, draw the circular top cap  if (topRadius != 0) {  angle = 0;  beginShape(TRIANGLE\_FAN);    // Center point  vertex(0, 0, 0);  for (int i = 0; i < sides + 1; i++) {  vertex(topRadius \* cos(angle), 0, topRadius \* sin(angle));  angle += angleIncrement;  }  endShape();  }    // If it is not a cone, draw the circular bottom cap  if (bottomRadius != 0) {  angle = 0;  beginShape(TRIANGLE\_FAN);    // Center point  vertex(0, tall, 0);  for (int i = 0; i < sides + 1; i++) {  vertex(bottomRadius \* cos(angle), tall, bottomRadius \* sin(angle));  angle += angleIncrement;  }  endShape();  }  } |
| **활동 사진** | *※ 활동하고 있는 모습, 회로도, 완성된 모습, 실행화면 등..*  <MPU6050 회로도 참고>    <결과 사진 참고> |
| **활동 결과** | *※ 활동 내용을 통해 나온 결과 정리*  MPU6050을 3D로 받아보았고, 제작할 때, GUI 부분에서 그래픽은 OpenGL. 파이썬으로 시도해 볼 생각이고, 그래프는 MATPLOTLIB라이브버리를 사용할 것입니다. 아두이노 부분은 개발도구를 아두이노 아이들을 GUI분은 비주얼 스튜디오 코드를 이용할 것입니다.  그리고 고도 그래프, 3D시각화, 돌아간 각도, 날짜 시간을 추가하고 다른임무가 있을 경우 창(카메라)등의 기능을 추가할 것 같습니다. |
| **계획** | *※ 다음 활동 시 해야할 일*  GUI에서 어떤 기능이 들어갈지 생각해 둔 부분과 배치도를 사용하여 추가해볼 계획입니다. |

2024년 5 월 12 일

작성자 : 장현주 (인)